REMARKS

This paper is responsive to the Final Office Action of December 26, 2008, in which claims 1-67 are currently pending. All of the pending claims stand rejected. By this response, claims 1, 17, 33 and 49 are amended. Claims 1-67 remain pending in the application.

At paragraph 3 of the Office Action, the Examiner rejects claims 1-4, 6-9, 11, 13-15, 17-20, 22-25, 27, 29-31, 33-36, 38-41, 43, 45-47, 49, 52-54 and 56-67 under 35 U.S.C. §102(e) as being anticipated by U.S. Patent No. 6,820,255 (Babaian). The Applicant has presented arguments with respect to Babaian in prior Responses. To expedite prosecution, the Applicant amends independent claims 1, 17, 33 and 49. Among those amendments, the Applicant includes the limitations of dependent claims 50 and 63, which have been canceled without prejudice. Support for these amendments can be found throughout the specification, for example at paragraphs [0207] and [0208]. These amendments are solely intended to expedite prosecution of the application and the Applicant reserves the right to pursue the subject matter of the original claims, such as in one or more continuation applications.

Babaian teaches that a software layer 206 contains translation processes 202-204 (including an interpreter 203, an optimizing binary translator 202 and a dynamic analysis process 204) that work together to translate foreign code into translated code that is executable by a host computer 102, and store the translated code into a code database 208. Then, later, the previously translated code is retrieved from the code database 208 for execution by the host computer 102, instead of performing another translation of the same sequence of foreign code. See Babaian at col. 3, lines 14-25 and col. 7, lines 40-67. To this end, Babaian seems to disclose no more than a "local cache" of the translated code to be later re-used by the same translation processes 202-204.

Notably, the software layer 206 of Babaian "functions as a basic operating system for the host platform" and "its primary function is to interface the binary translation system with hardware specific features of the host platform". See Babaian at col. 5, lines 51-55. All of the foreign code passes through this software layer 206 to execute on the host CPU 102. When a foreign application program or operating system is executed, foreign code is loaded into system memory from a disk drive or other storage medium under control of the software layer 206. The translated code is stored

and retrieved in the code database 208 in sequences of one page (4Kb) as a conveniently sized unit. See Babaian at col. 9, lines 12-22.

Therefore, Babaian does not disclose at least the following feature from claim 1:

> "providing a second translator instance which is different from the first translator instance and which translates the subject code of the second program into the target code, wherein the second translator instance operates simultaneously with the first translator instance;"

That is, claim 1 clearly recites that first and second translator instances operate simultaneously, and that each translate a respective program. There are no such "first and second translation instances" in Babaian. Indeed, this is admitted in the Office Action at page 16 (in relation to previous dependent claim 50).

Further, Babaian does not disclose at least the following feature from claim 1:

> "retrieving the cached portion of the target code from the shared code cache facility upon a compatibility detection between said cached portion of the target code and a second portion of the subject code in the second program [i.e. the cached target code produced from the first code portion in the first program is now detected as being compatible with this second code portion in the second program],
>
> including loading the portion of the target code in the shared code cache facility into a portion of memory which is shared amongst at least the first and second translator instances; and
>
> copying at least one part of the shared code cache facility to a private portion of memory associated with the second translator instance upon modification of the at least one part of the shared code cache facility by the second translator instance"

Here, the Office Action at page 10 (in relation to previous dependent claim 63) draws attention to Babaian at col. 10, lines 10-40. However, Babaian at col. 10, lines 10-40 only discusses that the software layer 206 "maintains correspondence between the selected foreign code and the translated binary code located in the database 208 by using either a disk indexing technique or a hash", and loads the relevant sequence of translated code into system memory when needed by the software layer 206. However, Babaian at col. 10 lines 10-40 completely fails to mention either "shared memory" or "private memory" or any equivalent thereto. Further, Babaian completely fails

to mention the steps of loading into shared memory and copying into private memory up on modification as recited in claim 1.

The specification for example at [0207] highlights this shared code caching technique where the same portion of target code is used by multiple translator instances running simultaneously, and further highlights that the combined use of shared memory and private memory significantly reduce overall physical memory usage of the multiple translator instances.

Also, contrary to the rejections in the Office Action, Babaian does not disclose all of the limitations in the other independent claims. For example in claim 17 see:

> "provide a second translator instance which translates the subject code of a second program into the target code, wherein the second translator instance is different from the first translator instance and operates simultaneously with the first translator instance, and wherein the second translator instance retrieves the cached portion of the target code from the shared code cache facility upon a compatibility detection between said cached portion of the target code and a second portion of the subject code in the second program, including loading the portion of the target code in the shared code cache facility into a portion of memory which is shared amongst at least the first and second translator instances; and copy at least one part of the shared code cache facility to a private portion of memory associated with the second translator instance upon modification of the at least one part of the shared code cache facility by the second translator instance."

For example in claim 33 see:

> "translator code for ... providing a second translator instance executing on the computer, wherein the second translator instance is different from the first translator instance and wherein the second translator instance translates the subject code of the second program into the target code, including retrieving the cached portion of the target code from the shared code cache facility upon a compatibility detection between said cached portion of the target code and a second portion of the subject code in the second program, including loading the portion of the target code in the shared code cache facility into a portion of memory which is shared amongst at least the first and second translator instances, and copying at least one part of the shared code cache facility to a private portion of memory associated with the second translator instance upon modification of the at least one part of the shared code cache facility by the second translator instance".

For example in claim 49 see:

> "program code for caching said portion of target code into a shared code cache facility and for providing a second translator instance for retrieving said target code from said shared code cache facility upon detection of compatibility between a second portion of subject code in a second program and said portion of target code, wherein said second translator instance operates simultaneously with the first translator instance; and program code for loading the portion of the target code in the shared code cache facility into a portion of memory which is shared amongst at least the first and second translator instances, and copying at least one part of the shared code cache facility to a private portion of memory associated with the second translator instance upon modification of the at least one part of the shared code cache facility by the second translator instance".

We respectfully submit that the claimed subject matter is not taught by Babaian. Further, none of the other cited documents fulfill the above-mentioned deficiencies in Babaian. Instead, the claimed subject matter is novel over Babaian and is not obvious to one having ordinary skill in the art. Accordingly, independent claims 1, 17, 33 and 49 as amended should be allowable.

Also, we respectfully disagree with the comments and observations in the Office Action concerning the dependent claims with respect to Babaian. These claims are all allowable not least because they depend from allowable independent claims.

At paragraph 5, the Examiner rejects claims 5, 21 and 37 under 35 U.S.C. §103(a) as being unpatentable over Babaian in view of U.S. Patent No. 6,826,750 (Curtis) and in further view of U.S. Patent No. 6,249,788 (Ronstrom). At paragraph 7, the Examiner rejects claims 10, 26 and 42 under 35 U.S.C. §103(a) as being unpatentable over Babaian in view of Miller. At paragraph 8, the Examiner rejects claims 12, 16, 28, 32, 44 and 48 as being unpatentable over Babaian in view of U.S. Patent No. 5,475,840 (Nelson). However, neither Curtis, Ronstrom, Miller nor Nelson, alone or in combination, supplies that which is missing from Babaian as set forth above. Therefore those claims should be allowable as depending from allowable base claims.

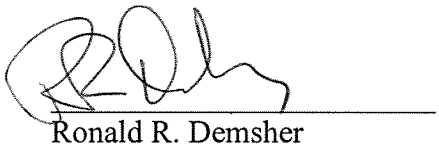
In view of the above amendment, applicant believes the pending application is in condition for allowance.

Filed herewith is a Request and fee for a One-Month Extension of Time, which extends the statutory period for response to expire on April 26, 2009. Accordingly, Applicant respectfully

submits that this response is being timely filed. Applicant believes no other fee is due with this response. However, if a fee is due, please charge our Deposit Account No. 08-0219, under Order No. 1801270.00140US1 from which the undersigned is authorized to draw.

Respectfully submitted,

Dated: April 27, 2009



Ronald R. Demsher
Registration No.: 42,478
Attorney for Applicant(s)

Wilmer Cutler Pickering Hale and Dorr LLP
60 State Street
Boston, Massachusetts 02109
(617) 526-6000 (telephone)
(617) 526-5000 (facsimile)